AMD

Power Management Resume Support for AMD-768™ Peripheral Bus Controller

Application Note

Publication # **25818** Rev: **1.01**
Issue Date: **February 2002**

Revision History

Date	Rev	Description
February 2002	1.01	Initial public release. Incorporated edits from legal department.
December 2001	1.0	Initial NDA release.

Introduction

This application note provides information for implementing the wake-up event in S1 and S3 states on an AMD Athlon™ processor system using an AMD-768™ peripheral bus controller. The controller is designed to support various mechanisms that allow systems to resume from ACPI Low Power states. This application note describes software BIOS support required for supporting the ACPI S1 (Power on Suspend) and S3 (Suspend to RAM) Low Power states.

The mechanisms described in this application note are supported in the Microsoft® Windows® operating systems.

Hardware Requirements

This application note describes the BIOS support required for using the SMSC (LPC61W492) or Winbond (W83627HF/F) Super I/O controllers in conjunction with the AMD-768 peripheral bus controller. The connections between the Super I/O controller and the AMD-768 peripheral bus controller are documented in the latest AMD-760™ MPX Chipset Motherboard Change List. Other Super I/O controllers may have similar register sets, but they are not discussed in this application note.

System designers using different Super I/O controllers may need to employ different BIOS and board-level designs in order to successfully implement ACPI S1 and S3. They should review this application note in detail to understand the issues and contact AMD for additional support if necessary.

Implementation

Wake Up from Keyboard and Mouse When Suspended in Either S1 or S3 States

Use of the ACAV monitoring function available on the AMD-768 peripheral bus controller is recommended, with keyboard and mouse monitoring functions provided by the Super I/O controller. This application note provides an example using the SMSC LPC61W492 or a Winbond W83627HF/F IO controller. The SIO PME pin is used as the wake-up event for S1 state and the PSOUT pin is used for S3 state.

The wake-up event functions provided by the SIO_PME and PSOUT pins must be enabled just before entering the S1 and S3 states, respectively. These functions must be disabled once the system resumes to normal operation. It is recommended that BIOS enable/disable these pins using SM mode. In an ACPI system, an SMI should be generated when a sleep command is issued to the Sleep Command register in the AMD-768 peripheral bus controller. The BIOS is able to recognize the sleep type and enable the corresponding wake-up event prior to placing the system into the Sleep state.

High-Level System Algorithmic Process for ACPI S1 Support

The following describes interactions between the operating system, system BIOS, and underlying hardware necessary to correctly implement S1 support:

1. The end user requests that the system enter Standby mode via the appropriate control panel.
2. A BIOS SMI routine traps the operating system request to enter the Sleep mode.
3. Registers within the Super I/O device and AMD-768 peripheral bus controller are programmed to enable appropriate wake-up events.
4. BIOS SMI handler places the system in the Standby mode.
5. A PS/2 device event (keyboard or mouse) occurs to awake the system from standby.
6. The processor resumes execution within the SMI handler at the point just after that handler placed the system into Standby mode.
7. Registers within the Super I/O device and the AMD-768 peripheral bus controller are programmed to disable the wake-up events that were programmed in step 3.
8. The SMI handler exits and returns control to the operating system.
9. System returns to the Full On state.

High-Level System Algorithmic Process for ACPI S3 Support

The following describes interactions between the operating system, system BIOS, and underlying hardware necessary to correctly implement S3 support:

1. The end user requests that the system enter Standby mode through the appropriate control panel.
2. A BIOS SMI routine traps the operating system request for a hardware Sleep state.
3. Registers within the Super I/O device and AMD-768 peripheral bus controller are programmed to enable appropriate wake-up events.
4. The BIOS SMI handler saves required chipset and Super I/O register values in system memory.
5. The BIOS SMI handler places the system in the Standby mode.
6. A PS/2 device event (keyboard or mouse) occurs to awake the system from standby.
7. System returns from ACPI S3 state and processor starts from reset vector.
8. Chipset and Super I/O registers are restored in early BIOS POST sequence.
9. The PSOUT function is disabled in the S3 resume process.
10. The BIOS returns system control back to the operating system.
11. System returns to the Full On state.

Enable ACAV Monitoring Function

The ACAV (AC change detection) monitoring function is enabled by setting the AMD-768 peripheral bus controller ACAV_EN bit (bit 12 of PM22). The ACAV# pin is powered by VCC_AUX when in either the S1 or S3 state.

If the ACAV# pin is pulled low when in the Sleep state, the ACAV_STS bit in the AMD-768 peripheral bus controller (bit 12 of PM20) is set and the system transitions from the Sleep state to the full on state. The ACAV_EN bit can be enabled in the POST sequence and remains set all the time or just before setting the system into the Sleep state. The ACAV_STS bit must be cleared prior to returning control to the operating system when resuming operation from the S1 or S3 states. This is done by writing a “1” to the status bit from S1 or S3 state.

Enable SMI Trapping on Sleep Command

The AMD-768 peripheral bus controller is capable of generating an SMI on any sleep command. This is accomplished by setting both the SLPCMD_EN bit (bit 0 of PM32) and the SLP_EN bit (bit 5 of PM05). The SLPCMD_STS bit (bit 0 of PM30) becomes set when the SLP_EN bit is set.

The SLPCMD_EN bit should be set in the generic power management initialization process in the POST sequence and remains set at all times.

The BIOS SMI handler may determine that the SMI was generated by a sleep command by checking the SLPCMD_STS bit. The SLPCMD_STS bit must be cleared by writing a "1" to the bit in the handler. The sleep command may be determined from bits [4:2] of the Sleep Command register (PM05).

The following illustrates an example of the code.

```
SMI_handler:
      …….
      ;check if this SMI was generated from the Sleep Command
      MOV     dx,   PM30      ;offset 30h of PMIO register
      IN      al,   dx        ;get SMI status
      TEST    al,   1         ;is SLPCMD_STS set?
      JNZ                     Sleep_CMD_handler;yes,
process sleep command
      …….
      …….
      Sleep_CMD_handler:
      ;clear SLPCMD_STS bit
      MOV     al, 1           ;write a “1” to SLPCMD_STS
      OUT     dx, al          ;
      ;clear SLPCMD_EN bit
      MOV     dx, PM32        ;offset 32h of PMIO register
      IN      al, dx          ;
      AND     al, 0FEh        ;
      OUT     dx, al          ;clear SLPCMD_EN bit
      ;get the sleep type from sleep command register
```

```
MOV     dx, PM05        ;Sleep Command register
IN      al, dx          ;
MOVZX   bx, al
AND     bx, 1Ch         ;get Sleep Command bits, bits 4-2
;use sleep type value as an offset to get the sleep handler
; address
SHR     bx, 1           ;offset to the entry of the
; handler table
ADD     bx, offset S_handler_tbl
MOV     bx, cs:[bx]
JMP     bx
……
……

S0_handler:
……
;set system to FON mode
MOV     dx, PM05        ;sleep command register
MOV     al, 20h         ;SLP_EN=1, SLP_TYPE=000b (FON)
OUT     dx, al          ;send FON command
JMP     SMI_exit

S1_handler:
;process S1 related task
……

;enable S1 wake-up event, such as SIO PME for keyboard and
;mouse interrupt
……
……

;set system to S1 state
MOV     dx, PM05        ;sleep command register
MOV     al, 24h         ;SLP_EN=1, SLP_TYPE=001b (POS)
OUT     dx, al          ;send POS command

;system stops here…
XOR     cx, cx          ;
LOOP    $

;system wakes up from pre-defined wake-up event, such as
;keyboard and mouse event

;disable S1 wake-up event, such as SIO PME for keyboard and
;mouse interrupt
……
……

;clear ACAV_STS bit
MOV     dx, PM20        ;offset 20h of PMIO register
MOV     ax, 1000h       ;bit 12
```

```
OUT     dx, ax          ;clear ACAV_STS

;re-enable SLPCMD_EN bit
MOV     dx, PM32        ;offset 32h of PMIO register
IN      al, dx          ;
OR      al, 1           ;set SLPCMD_EN bit
OUT     dx, al          ;
JMP     SMI_exit

S3_handler:
;process S3 related task
……

;enable S3 wake-up event, such as PSOUT for keyboard and
;mouse interrupt
……

;enable SDRAM self-refresh mode
……

;set system to S3 state (STR)
MOV     dx, PM05        ;sleep command register
MOV     al, 34h         ;SLP_EN=1, SLP_TYPE=101b (STR)
OUT     dx, al          ;send STR command

;system stops here…..
XOR     cx, cx
LOOP    $
S4_handler:
;process S4 related task
……
;set system to S4 state (STD)
MOV     dx, PM05        ;sleep command register
MOV     al, 38h         ;SLP_EN=1, SLP_TYPE=110b (STD)
OUT     dx, al          ;send STD command

;system stops here
XOR     cx, cx
LOOP    $

S5_handler:
;set system to S5 state (SOFF)
MOV     dx, PM05        ;sleep command register
MOV     al, 3Ch         ;SLP_EN=1, SLP_TYPE=111b (SOFF)
OUT     dx, al          ;send SOFF command

;system stops here
XOR     cx, cx
LOOP    $
Rsvd_handler:
JMP     SMI_exit        ;do nothing
```

```
SMI_exit:
RSM                     ;simplified example for
                        ; exiting SM mode
S_handler_tbl:
DW      offset S0_handler      ;NOTE: the offset of
DW      offset S1_handler      ;  each handler has to
DW      offset rsvd_handler    ;  be adjusted according
DW      offset rsvd_handler    ;  to where the SMI
DW      offset rsvd_handler    ;  handler is installed,
DW      offset S3_handler      ;  with consideration of
DW      offset S4_handler      ;  EIP been normized at
DW      0ffset S5_handler      ;  at the entry of SMI mode.
```

Enable PME Pin for Keyboard and Mouse Wake Up Event in S1 State

To allow the system to wake from S1 state through the keyboard or mouse, the Super I/O PME pin must be enabled in the Super I/O controller. This is done before entering the S1 state within the SMI handler routine.

The following sample code shows how this can be accomplished.

```
;enter the configuration mode
MOV     dx, SIO_CFG_ADDR;2Eh or 4Eh depending on
                        ; system implementation
MOV     al, 087h        ;config mode command
OUT     dx,al
NOP
NOP
OUT     dx,al

;select device A
MOV     dx, SIO_CFG_ADDR
MOV     al, 7
OUT     dx, al
MOV     al, 0ah         ;select device A
INC     dx              ;SIO data port
OUT     dx, al
DEC     dx

;enable device A
MOV     al, 30h         ;register 30h
OUT     dx, al
MOV     al, 1           ;set enable bit
INC     dx
OUT     dx, al
DEC     dx

;clear keyboard and mouse status
MOV     al, 0f3h        ;register 0f3h
```

```
OUT     dx, al
MOV     al, 30h         ;clear bits 5:4
INC     dx
OUT     dx, al
DEC     dx

;enable keyboard and mouse interrupt for SMI or PME
MOV     al, 0f6h        ;register 0f6h
OUT     dx, al
MOV     al, 30h         ;set bits 5:4
INC     dx
OUT     dx, al
DEC     dx

;enable event generation from PME
MOV     al, 0f9h        ;register 0f9h
OUT     dx, al
MOV     al, 5           ;set bits 2 and 0
INC     dx
OUT     dx, al
DEC     dx

;exit configuration mode
MOV     al, 0aah        ;exit command
OUT     dx, al
```

The SIO PME function for keyboard and mouse interrupts must be disabled upon waking up from the S1 state. The following sample code shows how this can be accomplished.

```
;enter the configuration mode
MOV     dx, SIO_CFG_ADDR;2Eh or 4Eh depending on
                        ;system implementation
MOV     al, 087h        ;config mode command
OUT     dx, al
NOP
NOP
OUT     dx, al

;enable access to device A
MOV     dx, SIO_CFG_ADDR
MOV     al, 7
OUT     dx, al
MOV     al, 0ah         ;select device A
INC     dx              ;SIO data port
OUT     dx, al
DEC     dx

;enable device A
MOV     al, 30h         ;register 30h
OUT     dx, al
MOV     al, 1           ;set enable bit
```

```
INC     dx
OUT     dx, al
DEC     dx

;disable event generation for PME
MOV     al, 0f9h        ;register 0f9h
OUT     dx, al
MOV     al, 0           ;clear all bits
INC     dx
OUT     dx, al
DEC     dx

;disable keyboard and mouse interrupt for SMI or PME
MOV     al, 0f6h        ;register 0f6h
OUT     dx, al
MOV     al, 0           ;clear all bits
INC     dx
OUT     dx, al
DEC     dx

;clear keyboard and mouse status
MOV     al, 0f3h        ;register 0f3h
OUT     dx, al
MOV     al, 30h         ;by set bits 5:4
INC     dx
OUT     dx, al
DEC     dx

;disable device A
MOV     al, 30h         ;register 30h
OUT     dx, al
MOV     al, 0           ;clear enable bit
INC     dx
OUT     dx, al
DEC     dx

;exit configuration mode
MOV     al, 0aah        ;exit command
OUT     dx, al
```

Enable PSOUT Pin for Keyboard and Mouse Wake-Up Events When Entering the S3 State

The PSOUT pin remains powered when the system is in the S3 state. To allow the system to wake from the S3 state through keyboard or mouse events, the PSOUT pin must be enabled *before* the system enters the S3 state. This is done in the S3_handler.

The following sample code shows how this can be accomplished.

```
;enter the configuration mode
MOV     dx, SIO_CFG_ADDR ;2Eh or 4Eh depending on
                         ;system implementation
MOV     al, 087h         ;config mode command
OUT     dx, al
NOP
NOP
OUT     dx, al

;enable access to device A
MOV     dx, SIO_CFG_ADDR
MOV     al, 7
OUT     dx, al
MOV     al, 0ah          ;select device A
INC     dx               ;SIO data port
OUT     dx, al
DEC     dx

;enable device A
MOV     al, 30h          ;register 30h
OUT     dx, al
MOV     al, 1            ;set enable bit
INC     dx
OUT     dx, al
DEC     dx

;clear keyboard and mouse wake-up status
MOV     al, 0e3h         ;register 0e3h
OUT     dx, al
INC     dx
IN      al, dx           ;read to clear status
DEC     dx

;enable keyboard and mouse wake-up via PANSW_OUT (PSOUT)
MOV     al, 0e0h         ;register 0e0h
OUT     dx, al
INC     dx
MOV     al, 0e3h
OUT     dx, al
DEC     dx

;exit configuration mode
MOV     al, 0aah         ;exit command
OUT     dx, al
```

The PSOUT function must be disabled when resuming from the S3 state. The processor begins execution from the reset vector when transitioning from S3 to Full On state. The PSOUT function can be disabled when the Super I/O controller is reprogrammed with its initial values by the BIOS initialization sequence.

Reference Documents

The following documents provide additional information regarding the operation of the AMD-768 peripheral bus controller:

- *AMD-762™ System Controller Data Sheet*, order# 24088.
- *AMD-768™ Peripheral Bus Controller Data Sheet*, order# 24467.
- *AMD Athlon™ System Bus Specification*, order# 21902
- *AMD Athlon™ Processor BIOS, Software, and Debug Tools Developers Guide*, order# 21656
- *Power Management Resume Support for AMD-768™ Peripheral Bus Controller Application Note*, order# 25818
- *Motherboard Design Guide for the AMD-761™ and AMD-762™ System Controllers*, order# 24281